2018 13th IAPR International Workshop on Document Analysis Systems

Handwriting Recognition of Historical Documents with few labeled data

Edgard Chammas and Chafic Mokbel
University of Balamand
El-Koura, Lebanon
{*edgard,chafic.mokbel*}*@balamand.edu.lb*

Laurence Likforman-Sulem
Institut Mines Telecom, Telecom ParisTech and Université Paris-Saclay
Paris, France
laurence.likforman@telecom-paristech.fr

***Abstract*—Historical documents present many challenges for offline handwriting recognition systems, among them, the segmentation and labeling steps. Carefully annotated text-lines are needed to train an HTR system. In some scenarios, transcripts are only available at the paragraph level with no text-line information. In this work, we demonstrate how to train an HTR system with few labeled data. Specifically, we train a deep convolutional recurrent neural network (CRNN) system on only 10% of manually labeled text-line data from a dataset and propose an incremental training procedure that covers the rest of the data. Performance is further increased by augmenting the training set with specially crafted multi-scale data. We also propose a model-based normalization scheme which considers the variability in the writing scale at the recognition phase. We apply this approach to the publicly available READ dataset[1]. Our system achieved the second best result during the ICDAR2017 competition [1].**

***Keywords*-CRNN, handwriting recognition, historical documents, variability, multi-scale training, model-based normalization scheme, limited labeled data**

I. Introduction

Most state-of-the-art offline handwriting text recognition (HTR) systems work at the line level by transforming the text-line image into a sequence of feature vectors. These features are fed into an optical model (e.g, recurrent neural network) in order to recognize the handwritten text. Recent work on text detection and localization [2] at the document level, and joint line segmentation and recognition at the paragraph level [3] showed promising results. However, the best recognition results are still achieved by the systems working at the line level [4]. The automatic segmentation of paragraphs into lines is even more challenging on historical documents. Old manuscripts are often acquired as low resolution images with degraded quality, with overlapping characters across adjacent text-lines (see figure 1). Supervised (or at least semi-supervised) paragraph segmentation is needed to label each text-line in order to train an HTR system. However, this is a tedious and time consuming task that is not always feasible for different reasons (budget, time, priority, and availability of text data). When transcriptions are primarily provided at the paragraph level, the first challenge consists in aligning the training transcription data with the corresponding lines in the image. In this paper, we propose to perform such an alignment after training a first recognition system on a limited amount of annotated data. The first system serves to bootstrap the whole process. We also suggest to augment the amount of data by generating multiscale synthetic data in order to better consider the scale factor in the test images. We apply this approach to the READ dataset, a multilingual Latin offline handwriting dataset. The training data provided during the ICDAR2017 competition[2] were part of the Alfred Escher Letter Collection (AEC), with a large vocabulary of more than 130k words. The test data were letter documents from the same period of AEC. In section II, we present our state of the art deep convolutional recurrent neural network (CRNN) that we used in ICDAR2017 competition on handwritten text recognition. During the competition, 10000 pages were available for training with transcriptions provided at the paragraph level only. In section III, we demonstrate how to train an HTR system by using a small amount of manually segmented and labeled text-lines to create a bootstrap model. We further improve the performance of our system by augmenting the training set with specially crafted synthetic data, explicitly taking into consideration the variability in the writing scale (section IV). In section V, we propose a model-based normalization scheme that considers the writing scale variability in the test data. Our system achieved the second best result during the ICDAR2017 competition.

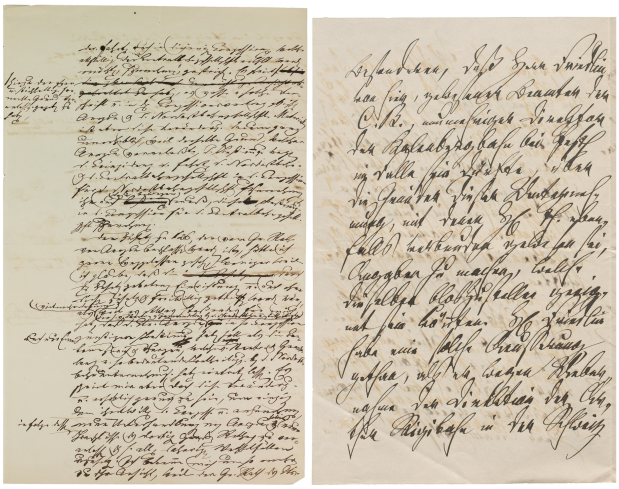

Figure 1: Old manuscripts from the READ 2017 dataset.

[1] https://read.transkribus.eu/

[2] https://scriptnet.iit.demokritos.gr/competitions/8/

978-1-5386-3346-5/18 $31.00 © 2018 IEEE
DOI 10.1109/DAS.2018.15

II. CRNN SYSTEM DESCRIPTION

Our system is a deep Convolutional Recurrent Neural Network (CRNN) inspired from the VGG16 architecture [5] used for image recognition. We use a stack 13 convolutional (3×3 filters, 1×1 stride) layers followed by three Bidirectional LSTM layers with 256 units per layer. Each LSTM unit has one cell with enabled peephole connections. Spacial pooling (max) is employed after some convolutional layers. To introduce non-linearity, the Rectified Linear Unit (ReLU) activation function was used after each convolution. It has the advantage of being resistant to the vanishing gradient problem while being simple in terms of computation, and was shown to work better than sigmoid and tanh activation functions [6]. A square shaped sliding window is used to scan the text-line image in the direction of the writing. The height of the window is equal to the height of the text-line image, which has been normalized to 64 pixels. The window overlap is equal to 2 pixels to allow continuous transition of the convolution filters. For each analysis window of 64×64 pixels in size, 16 feature vectors are extracted from the feature maps produced by the last convolutional layer and fed into the observation sequence. It is worth noting that the amount of feature vectors extracted from each sliding windows is important. The number must be reasonable as to provide a good sampling for the image. Based on previous experiments, we found out that oversampling (32 feature vectors per window) and under-sampling (8 feature vectors per window) will decrease the performance. Sixteen feature vectors were found to work best for our architecture. Since for each of the 16 columns of the last 512 feature maps, the columns of height 2 pixels are concatenated into a feature vector of size 1024 (512×2).

Thanks to the CTC objective function [7], the system is end-to-end trainable. The convolutional filters and the LSTM units weights are thus jointly learned within the back-propagation procedure. We chose to keep the network simple with a relatively small number of parameters. We thus combine the forward and backward outputs at the end of the BLSTM stack [8] rather than at each BLSTM layer. We also chose not to add additional fully-connected layers. The LSTM unit weights were initialized as per [9] method, which proved to work well and helps the network convergence faster. This allows the network to maintain a constant variance across the network layers which keeps the signal from exploding to a high value or vanishing to zero.

The weight matrix W_{ij} were initialized with a uniform distribution given as $W_{ij} \sim U(-\frac{\sqrt{6}}{n}, \frac{\sqrt{6}}{n})$, where n is the total number of input and output neurons at the layer (assuming all layers are of the same size).

Adam optimizer [10] was used to train the network with initial learning rate of 0.001. This algorithm could be thought of as an upgrade for RMSProp [11], offering bias correction and momentum [12]. It provides adaptive learning rates for the stochastic gradient descent update computed from the first and second moments of the gradients. It also stores an exponentially decaying average of the past squared gradients (similar to Adadelta [13] and RMSprop) and the past gradients (similar to momentum). Batch normalization as described in [14], was added after each convolutional layer in order to accelerate the training process. It basically works by normalizing each batch by both mean and variance. The network was trained in an end-to-end fashion with the CTC loss function [7]. A token passing algorithm was used for decoding [15]. It integrates a bigram language model with modified Kneser-Ney discounting [16], built from the available training data. It is worth noting that no preprocessing is needed. The system works directly on raw images. The full architecture is provided at the end of this paper (figure 5).

III. INCREMENTAL TRAINING WITH FEW LABELED DATA

With no line information provided, few labeled text-lines are needed to bootstrap the training process. We used an automatic segmentation algorithm to extract line images from the document images. The algorithm selects candidate baselines by analyzing contours distribution. It then assigns each contour to one of the baselines based on a number of criteria, related to the average distance between two lines and the distance between the contour center and the line (see figure 2). Only 10% of the pages were manually verified, making sure the line segmentation is correct, and used to bootstrap the training process. Besides the 10,000 training pages, 50 annotated pages at the line level were provided during the competition and were used for validation in the training process. The initial recognition system, trained on 10% of the data, achieved 9.2% raw label error rate (LER). This performance can be considered good enough to allow an incremental training of the network from the rest of the data.

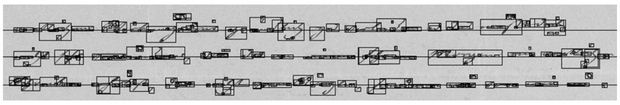

Figure 2: Candidate baselines with contours bounding boxes.

As a next step, the system was set to recognize the remaining 90% of the segmented line images in the training set. The recognized lines were mapped to lines in the ground-truth data for each page, based on the Levenshtein distance [17] between the text lines. A mapping is considered valid when the edit distance is less than or equal to half the length of the reference line. Following this process, and according to this threshold on the Levenshtein distance, 80% of the available text-lines were selected to retrain the system, while the rest (20%) were discarded. The retrained system achieved a relative decrease of 20% in raw LER on the validation set (see table I). The process could have been restarted after having trained the system with the new data, or even iterated. An improved recognition performance could have recovered more training lines. However, we have noticed that most of the discarded line images in the first iteration resulted from wrong segmentation (e.g., two text-lines in a single image, cropped text-line, etc), due to the fact that the algorithm is sensitive to the writing skew. Therefore, more advanced segmentation algorithms are needed to improve the selection/training process, like the ones based on Seam Carving technique [18] and dynamic programming, which would have resulted in fewer segmentation errors and therefore more labeled training data. The whole process can be summarized at the end of this paper (algorithm 1).

Table I: System performance on the validation set with different amount of training data.

System	Number of text-lines	Label Error Rate (LER)
10% training data	~20k	9.2%
80% training data	~160k	7.4%

IV. Integration of multi-scale training data

To further enhance the performance of the system, we exploited the variability in the writing scale to augment the training set with text-line images at multiple scales. Based on a vertical scale score [19], the training lines were first classified into 3 classes (Large, Medium and Small) via Jenks natural breaks optimization algorithm [20]. By dividing the training set over the three classes, the data volume per class become smaller. To address this problem, we expanded the training set for each class by adding synthetic data resulting from scaling the other classes data. For example, we reduce the large images and stretch the small ones (by a predetermined factor for each class) to expand the number of medium sized images. Or we reduce the medium and large sized images to extend the set of small images, etc. To calculate the scaling factors by which a certain image of a given scale class is enlarged or reduced, the average scale measurement score is calculated on the data. For instance, to transform an image I of class X to an image J of class Y, we scale I by $E(Y)/E(X)$, where E(X) and E(Y) are the average scale score values for class X and Y respectively. We retrained the baseline system on multi-scale data for one epoch and achieved a 6.5% raw LER; a relative improvement by 12% from the previous system.

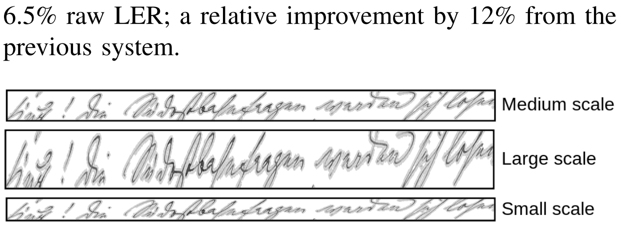

Figure 3: An example from the READ dataset where a text-line classified as Medium scale is transformed into a Large and Small scale versions.

V. Model-based normalization scheme

To further improve the performance, we proposed to consider the variability in the writing scale in a model-based normalization scheme, where the test data are equalized in order to best fit the core model. In general, consider the recognition phase where a test image characterized by a specific variability is provided at the input of a system trained on a general training set. According to the statistical decision theory, the recognition task identifies the most likely word sequence given the observations as:

$$\hat{s} = \arg\max_s Pr(s|\underline{\underline{X}}) \tag{1}$$

where s represents a word sequence, and $\underline{\underline{X}}$ the observation sequence. To cope with a variability factor θ in a test image, it is supposed that a transformation $T_\theta(.)$ exists with contextual parameter vector θ permitting to reduce this variability to a minimum. It is assumed that this parameter is hidden and cannot be measured. A normalized version of the input image $\underline{\underline{X}}$ can be defined as:

$$\hat{\underline{\underline{X}}} = T_\theta(\underline{\underline{X}}) \tag{2}$$

Assuming the contextual parameter vector θ belongs to a finite set, equation 1 can integrate the normalization defined in equation 2 to become:

$$\begin{aligned}\hat{s} &= \operatorname*{argmax}_s \sum_\theta Pr(s,\theta|\underline{\underline{X}}) \\ &= \operatorname*{argmax}_s \sum_\theta Pr(s|\theta,\underline{\underline{X}})Pr(\theta|\underline{\underline{X}}) \\ &= \operatorname*{argmax}_s \sum_\theta Pr(s|T_\theta(\underline{\underline{X}}))Pr(\theta|\underline{\underline{X}})\end{aligned} \tag{3}$$

For all possible normalizations of the input $\underline{\underline{X}}$, the system produces solutions with the corresponding scores, considered as posterior probabilities. A combination of the scores permits to re-select the optimal solution (see figure 4). This is considered as an approximation of the right-hand term of equation 3.

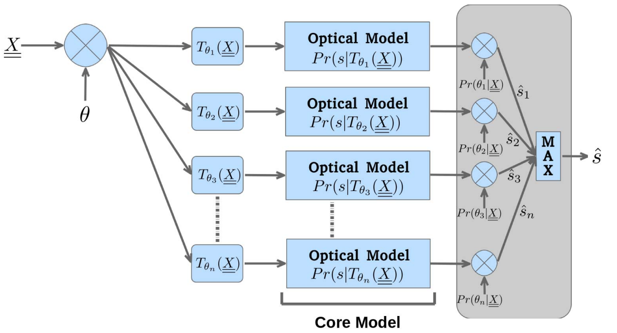
$\underline{\underline{X}}$
θ
$T_{\theta_1}(\underline{\underline{X}})$
$T_{\theta_2}(\underline{\underline{X}})$
$T_{\theta_3}(\underline{\underline{X}})$
$T_{\theta_n}(\underline{\underline{X}})$
Optical Model $Pr(s|T_{\theta_1}(\underline{\underline{X}}))$
Optical Model $Pr(s|T_{\theta_2}(\underline{\underline{X}}))$
Optical Model $Pr(s|T_{\theta_3}(\underline{\underline{X}}))$
Optical Model $Pr(s|T_{\theta_n}(\underline{\underline{X}}))$
$Pr(\theta_1|\underline{\underline{X}})$
$Pr(\theta_2|\underline{\underline{X}})$
$Pr(\theta_3|\underline{\underline{X}})$
$Pr(\theta_n|\underline{\underline{X}})$
$\hat{s}_1$
$\hat{s}_2$
$\hat{s}_3$
$\hat{s}_n$
MAX
$\hat{s}$
Core Model

Figure 4: Model-based normalization scheme.

We generated multiple versions of the test data by vertically scaling each text-line image to multiple scales (0.7, 0.8,..., 1.3). By considering equation 3, we could write:

$$\hat{s} = \operatorname*{argmax}_s \sum_{\theta=0.7}^{1.3} Pr(s|T_\theta(\underline{\underline{X}}))Pr(\theta|\underline{\underline{X}}) \tag{4}$$

We approximate equation 4 by the means of ROVER method [21]. The combination of the recognition scores of the different normalized versions of the test image has yielded to a relative improvement of 14% in WER from the baseline system. In Table II, we provide the word error rate (WER) and character error rate (CER) obtained with the different systems along with the result of the BYU (Computer Science Department) team who won the first place during the competition. The results show the significant increase in performance using the incremental training of our CRNN system. They also show a significant improvement when better considering the variability of writing scale. Finally, our best system achieves comparable results with the system ranked first in the contest. With 5.5% running OOV words [1], we believe the main difference in performance can be explained by our use of a bigram word language model. It is worth noting that our results can further be improved by using a more performant segmentation, which would also leads to more training data.

Table II: Effect of multi-scale data on the performance.

System	CER	WER
CRNN (1)	9.18%	25.07%
CRNN retrained with multi-scale data (2)	7.95%	23.09%
(2) + model-based normalization scheme	7.74%	21.58%
BYU System	7.01%	19.06%

VI. Conclusions and perspectives

In this work, we presented a state-of-the-art CRNN system for text-line recognition of historical documents. We showed how to train such system with few labeled text-line data. Specifically, we proposed to bootstrap an incremental training procedure with only 10% of manually labeled text-line data from the READ 2017 dataset. We also improved the performance of the system by augmenting the training set with specially crafted synthetic data at multi-scale. At the end, we proposed a model-based normalization scheme by introducing the notion of the variability in the writing scale to the test data. The combination of the multi-scale trained system results on multi-scale test data has yielded the best result. Our system achieved the second position in ICDAR2017 competition, with comparable performance to the winning system, while noting that the overall performance depends on both segmentation and recognition tasks. Our results can be improved by improving the segmentation algorithm which will permit to use more training data. Despite the complex network architecture, we noticed the large impact of the variability in the writing scale on the performance. As a future work, we will be looking into the possibilities for integrating this variability in the modeling. Possibly via an attention mechanism.

References

[1] J. A. Sánchez, V. Romero, A. H. Toselli, M. Villegas, and E. Vidal, "Icdar2017 competition on handwritten text recognition on the read dataset," in *Document Analysis and Recognition (ICDAR), 2017 14th International Conference on*. IEEE, 2017.

[2] B. Moysset, C. Kermorvant, and C. Wolf, "Full-page text recognition: Learning where to start and when to stop," *arXiv preprint arXiv:1704.08628*, 2017.

[3] T. Bluche, "Joint line segmentation and transcription for end-to-end handwritten paragraph recognition," in *Advances in Neural Information Processing Systems*, 2016, pp. 838–846.

[4] P. Voigtlaender, P. Doetsch, and H. Ney, "Handwriting recognition with large multidimensional long short-term memory recurrent neural networks," in *Frontiers in Handwriting Recognition (ICFHR), 2016 15th International Conference on*. IEEE, 2016, pp. 228–233.

[5] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[6] J. Gu, Z. Wang, J. Kuen, L. Ma, A. Shahroudy, B. Shuai, T. Liu, X. Wang, and G. Wang, "Recent advances in convolutional neural networks," *arXiv preprint arXiv:1512.07108*, 2015.

[7] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *Proceedings of the 23rd international conference on Machine learning*. ACM, 2006, pp. 369–376.

[8] A. Zeyer, R. Schlüter, and H. Ney, "Towards online-recognition with deep bidirectional lstm acoustic models." in *INTERSPEECH*, 2016, pp. 3424–3428.

[9] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, 2010, pp. 249–256.

[10] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[11] T. Tieleman and G. Hinton, "Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude," *COURSERA: Neural networks for machine learning*, vol. 4, no. 2, pp. 26–31, 2012.

[12] N. Qian, "On the momentum term in gradient descent learning algorithms," *Neural networks*, vol. 12, no. 1, pp. 145–151, 1999.

[13] M. D. Zeiler, "Adadelta: an adaptive learning rate method," *arXiv preprint arXiv:1212.5701*, 2012.

[14] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International Conference on Machine Learning*, 2015, pp. 448–456.

[15] A. Fischer, "Handwriting recognition in historical documents," *PhD diss*, 2012.

[16] S. F. Chen and J. Goodman, "An empirical study of smoothing techniques for language modeling," in *Proceedings of the 34th annual meeting on Association for Computational Linguistics*. Association for Computational Linguistics, 1996, pp. 310–318.

[17] V. I. Levenshtein, “Binary codes capable of correcting deletions, insertions, and reversals,” in *Soviet physics doklady*, vol. 10, no. 8, 1966, pp. 707–710.

[18] N. Arvanitopoulos and S. Süsstrunk, “Seam carving for text line extraction on color and grayscale historical manuscripts,” in *Frontiers in Handwriting Recognition (ICFHR), 2014 14th International Conference on*. IEEE, 2014, pp. 726–731.

[19] E. Chammas, C. Mokbel, and L. Likforman-Sulem, “Exploitation de léchelle décriture pour améliorer la reconnaissance automatique des textes manuscrits arabes,” *Document numérique*, vol. 19, no. 2, pp. 95–115, 2016.

[20] G. F. Jenks, “The data model concept in statistical mapping,” *International yearbook of cartography*, vol. 7, pp. 186–190, 1967.

[21] J. G. Fiscus, “A post-processing system to yield reduced word error rates: Recognizer output voting error reduction (rover),” in *Automatic Speech Recognition and Understanding, 1997. Proceedings., 1997 IEEE Workshop on*. IEEE, 1997, pp. 347–354.

Algorithm 1 Incremental alignment process

```
Require: $TrainSet$: Set of all training pages
Require: $RefText$: Ground-truth text paragraph for each page
  for each page $P$ in $TrainSet$ do
      $Lines[] \leftarrow Segment(P)$
      $RefLineIndex \leftarrow 0$
      for each line $L$ in $Lines$ do
          $RecSeq \leftarrow Recognize(L)$
          while $RefLineIndex < length(RefText[P])$ do
              $RefSeq \leftarrow RefText[P][RefLineIndex]$
              $EditDistance \leftarrow Levenshtein(RecSeq, RefSeq)$
              if $EditDistance < 0.5 \times length(RefSeq)$ then
                  $Map(L, RefSeq)$
                  $RefLineIndex \leftarrow RefLineIndex + 1$
              end if
          end while
      end for
  end for
```

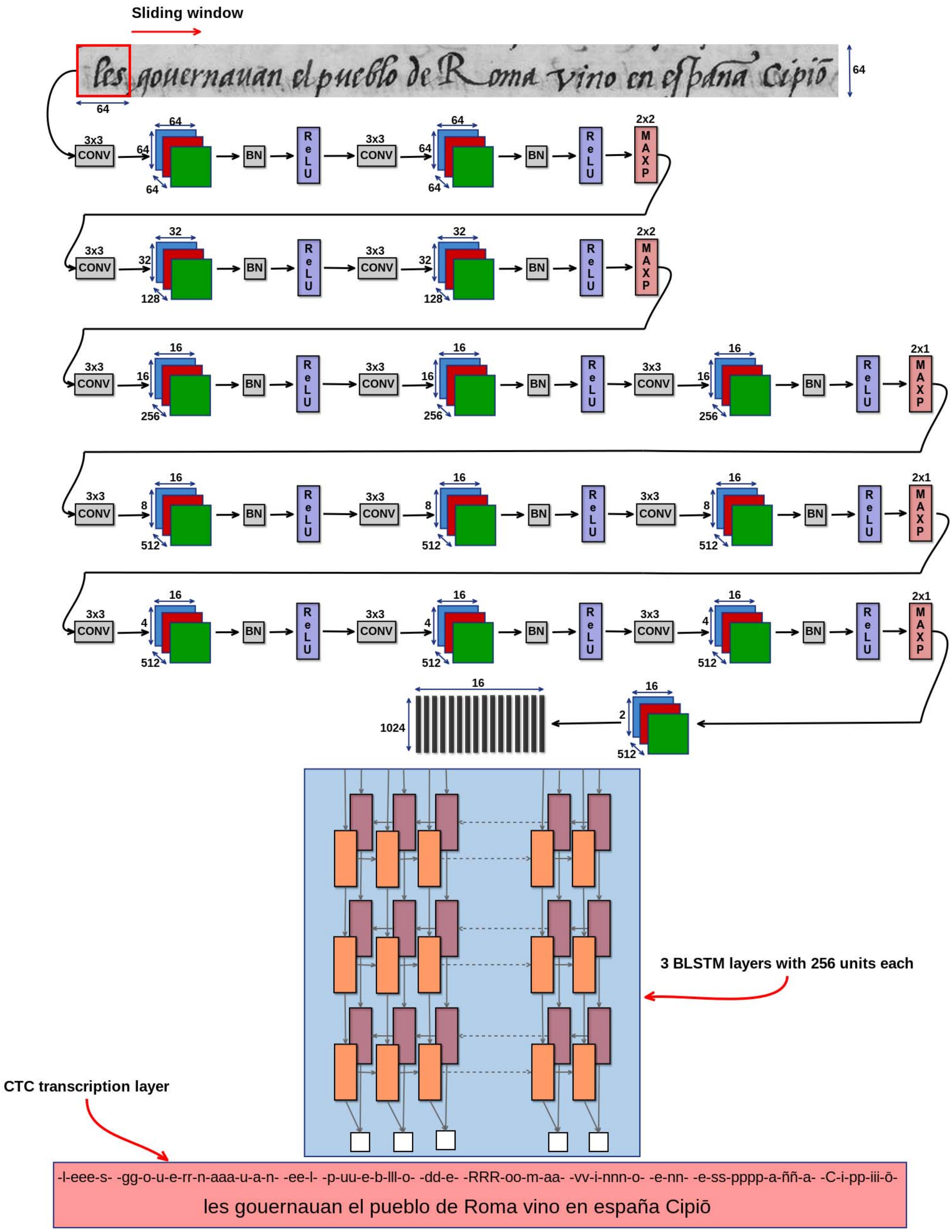

Figure 5: Recognition system.